ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

Indexing For Sequential Stochastic Mechanism-Based Movement Replicas In Recognition Of Human Activity

G.Karthika[1], V.Bhuvaneswari[2]

Second year M.E. (CSE), Sri Vidya College of Engineering and Technology, Virudhunagar, Tamilnadu, India[1]

Assistant Professor, CSE, Sri Vidya College of Engineering and Technology, Virudhunagar, Tamilnadu, India[2]

ABSTRACT— Today, numerous applications require the ability to monitor a continuous stream of fine-grained data for the occurrence of certain high-level activities. A number of computerized systems—including ATM networks, web servers, and intrusion detection systems—systematically track every atomic action they perform, thus generating massive streams of timestamped observation data, possibly from multiple concurrent activities. In this paper, address the problem of slowly indexing large number of observations and also the identification problem. A solution to this important problem would greatly benefit a broad range of applications, including fraud detection, video surveillance, and cyber security. In Existing work, tMAGIC has to be used for indexing activities and also used tMAGIC-evidence for evidence problem.We propose a Active merge algorithm to solve time complexity problem and sequence algorithm to solve identification problem.

KEYWORDS- Activity detection, indexing, stochastic automata, and timestamped data

I. INTRODUCTION

There are numerous applications where we need to monitor whether certain (normal or abnormal) activities are occurring within a flow of transaction data. For example, an online shop might want to monitor the activities occurring during a remote login session on its Web site in order to either better help the user or to identify users engaged in suspicious activities. In existing work, Index structure called Temporal MultiActivity Graph Index Creation (tMAGIC) to monitor multiple activities concurrently. tMAGIC starts by merging multiple temporal stochastic automata (each representing an activity) into a special kind of graph. It solves evidence problem and resolves space complexity. But it does not solve time complexity and identification problem.

Active Merge algorithms are a family of algorithms that run sequentially over multiple sorted lists, typically fastly producing more sorted lists as output. This is well-suited for machines with tape drives. Use has declined due to large random access memories, and many applications of merge algorithms have faster alternatives when a random access memory is available. It provides faster and efficient sorted list. There is no separate time calculation for all subactivities.Instead of, on which sub activity has to be purposeful to achieve the target that sub activities time calculation has to be calculated. Computation power is reduced. A sequence algorithm is an algorithm that takes one or more linear sequences as inputs. This algorithm has to get input as sequential manner so it has to solve the identification problem. It has to avoid inputs randomly and also it has to improve the quality of matching of values. It works well for both large differences and small differences of time values. In Section3 deals with the basic terms used in the paper such as Stochastic process with example, Temporal stochastic activity model with example,tMAGIC,Evidence problem Identification problem ,Observation table and Context information. Section paper 4 deals with the Evidence and identification problems. Section paper 5 deals with the Temporal MultiActivity Graph Index. Section 6 deals with Active merge algorithm Section 7 deals with the Sequence algorithm and Section 8 deals with the conclusion.

II. RELATED WORK

Limitations of traditional database management systems in supporting streaming applications and event processing have prompted extensive research in Data Stream Management Systems (DSMSs). Early yet comprehensive surveyof relevant issues in data stream management were presented in [8]. Amongst the several systems resulting from research efforts in this direction, of particular relevance is TelegraphCQ [6], a streaming query processor that filters, categorizes, and aggregates flow records according to one or more CQL [4] continuous queries, generating periodic reports. Differently from traditional queries on static data collections, results of continuous queries on streaming data need to be periodically and incrementally updated as new data is received. Although the system we propose in this paper operates on streams of observation data, the scope of our work is drastically different from the scope of DSMSs.

The aim of past work on indexing of activities was merely to retrieve previously recognized activities, not to recognize new ones. Such work includes that of Ben-Arie et al. [5] who use multidimensional index structures to store body pose vectors in video frames. Kerkez [10] develops indexes for case-based plan recognition where knowledge about planning situations enables the recognizer to focus on a subset of the plan library containing relevant past plans. A two-level indexing scheme, along with incremental construction of the plan libraries, is proposed to reduce the retrieval efforts of the recognizer.

In conclusion, our work differs from previous efforts by providing a algorithm to index and update observations in a data structure that solves both time and space complexity and also solve evidence and identification problems.

III. FRAMEWORK OVERVIEW

Transition caused by an event and results in movement from one state to another state. State represents snapshot of the system at some fixed point in time. Activity takes some amount of time duration culminates in an event. For ATM example->service completion.

Definition 3.1 (Stochastic process) If systems is stochastic because the internal system is affected by the random values of the external system.

Example 1. (Stochastic process)

- Automated Teller machine.
- Printed circuit board assembly operation.
- Runaway activity at airport.

Definition 3.2 (Temporal Stochastic Activity)A temporal stochastic activity (or just activity) is a labeled graph (V, E, δ) where,

- V is a finite set of observations
- E is a subset of (V×V)
- v Є V s.t. β v'ЄVs.t. (v',v) ЄE, i.e., there exists at least one start node in V ;
- v Є V s.t. β v'ЄVs.t. (v, v') ЄE, i.e., there exists at least one end node in V ;
- β v ЄV s.t. (v,v) ЄE, i.e., no self-loops are allowed;. δ:E ->Ω is a function that associates a timespan distribution with each edge in the graph, such that

  $\forall_{v \in V} \sum v' \in V|(v, v') \in E^{s(}\delta(v,v;))=1$

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

If A = (V, E,δ) is an activity and v€ V is a node,A.$p_{max(v)}$ denotes the maximum product of probabilities on any path in A from v to an end node.

Example 2. Fig. 1 shows an example temporal stochastic activity modeling a bill payment process in an onlinebanking system. A user will first access her accountspage (goAccounts) and either check her balance (checkBalance) or continue directly to the bill payment page (goBillpay). Assuming a time granularity of minutes, the edges between go Accounts and its successors are interpreted as in

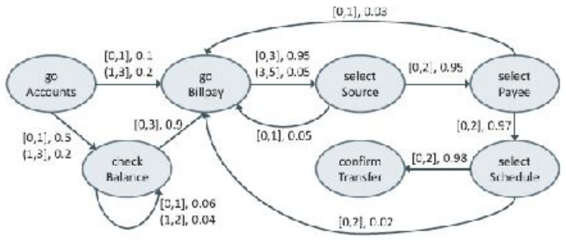

Fig.1 Example of temporal stochastic activity.

Definition 3.3 (Temporal stochastic activity model) This model represents the activity is uncertain and it changes over time.

Definition 3.4 (tMAGIC) Index structure called Temporal Multiactivity Graph Index Creation (tMAGIC) to monitor multiple activities concurrently. tMAGIC starts by merging multiple temporal stochastic automata (each representing an activity) into a special kind of graph. As observations arrive, the tMAGIC index can be updated either through individual insertions or through bulk insertions. The tMAGIC index tracks information about which nodes are start/end nodes for the original activities. For each node, it also stores 1) the maximum probability of reaching an end node for each activity in A, 2) a table that tracks partially completed activity occurrences where each record points to a tuple whose observation is part of the corresponding activity instance, as well as to the previous and successor records.

Definition 3.5 (Evidence problem) It tries to find all occurrences of an activity (with probability over a threshold) within a given sequence of observations.

Definition 3.6 (Identification problem) It tries to find the activity that best matches a sequence of observations.

Definition 3.7 (Observation table) In a Observation table, model a sequence of observations, that are observed in real time as well as a previously collected database of observations as continuously updated.

IV. EVIDENCE AND IDENTIFICATION PROBLEMS

This section formalizes the Evidence and Identification problems. Without loss of generality, we assume that observations are stored in a single relational observation table, denoted D. Each tuple t € D corresponds to a single observation, denoted t.obs, which is observed at a given time, denoted t.ts. When our framework is used for real time activity detection, our proposed insertion algorithm processes each observation as it is received, updates the index, and stores a tuple in the observation table. Conversely, when the framework is used to detect activities in a previously acquired body of data, our bulk insertion algorithm can pull all the observation tuples from the table and build the whole index.

Additionally, in some applications, each observation may be associated with context information (e.g., IP address, full name, spatial location), which might help discriminate between observations belonging to different activity occurrences. However, we do not assume this information to be available in general. For instance, in an intrusion detection

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT’14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

system, multiple attackers engaged indifferent activities, may need to perform some common steps, and they may appear to come from the same origin if they use proxies to conceal their real identities.

Definition 4.1 (Minimal Span Restriction). An activity occurrence D1 D is said to satisfy the minimal span restriction if and only if there is no other occurrence D2 Dof the same activity such that span(D2) span(D1)and prob(D1)≤ prob(D2)..

Definition 4.2 (Earliest Action Restriction). An activity occurrence {t1; . . . ; tn} D is said to satisfy the earliest action restriction if and only if Ai€[2,n] βw_i € D such that wi:ts < ti:ts and{t1; . . . ; ti-1; wi; ti+1; . . .; tn}is an occurrence of the same activity

Definition 4.3 (Evidence Problem). Given a temporal observation table D, a set of activities A, a time interval, [tss, tse] and a probability threshold p_t, compute all the occurrences D _ D of activities in A such that D occurs within the interval [tss,tse] and prob($D^{*)}$ ≥p_t.

Definition 4.4 (Identification Problem). Given a temporalobservation table D, a set of activities A, and a time interval [tss; tse], find the activity which occurs in D in the interval [tss; tse] with maximal probability among the activities in A.A solution to the identification problem could be biased because short activity occurrences generally tend to have higher probabilities. To remedy this, we normalize occurrence probabilities as defined in [2], by introducing the relative probability p^* of an occurrence D^* of activity A as $p^*(D^*) =^{prob\ (}D^{*)\ -p}{}_{min/}p_{max}\text{-}p_{min}$ where${}^{p}{}_{min}$ p_{max} are the lowest and highest possible probabilities of any occurrence of A.

V. TEMPORAL MULTIACTIVITY GRAPH INDEX

In order to monitor an observation table for occurrences of multiple activities, we first merge all temporal activity definitions from A={A_1……….A_k} into a single graph. We use id(A) to denote a unique identifier for activity A and I_A to denote the set {id($A_{1)}$...................id(A_k)}

Definition 5.1 (Temporal Multiactivity Graph Index). Let A ={A_1……….A_k} be a set of stochastic activities, where Ai =((V_i,E_i,δ_i), and let G=(VG, IA, δ_G) be the temporalmultiactivity graph built over A. A Temporal MultiactivityGraph Index is a 6-tuple IG = (δ$G_,$ $start_G$, end_G, max_G, tablesG, $completed_{G)}$ where $start_G$: VG -> $_2$ IA is a function that associates with each node v € VG, the set of activity ids for which v is a start node;

end_G: $VG_{2\text{->}}$ IA is a function that associates with each node v € VG, the set of activity ids for which v is an end node;

max_G: VG ×IA ->[0,1] is a function that associates with each pair (V,id(A_i)) the probability Ai.p_{max}(v), if v € Vi, and 0 otherwise;

For each v € VG, $tables_G$(v) is a set of records of the form (current, activityID, t_0, prob, previous, next), where current is a reference to an observation tuple,activityI D € 2 IA is an activity id, t0 €T is a timestamp, prob € [0,1], previous is a reference and next a set of references to records in $tables_G$

$completed_G$: IA -> 2^P, where P is the set of references to records in $tables_G$ is a function that associates with each activity identifier id(A) a set of references to records in $tables_G$ which correspond to completed instances of activity A.

Note that G, $start_G$, end_G, $tables_G$ can be computed a-priori,based on the set A of activities of interest. All tables that are part of the index ($tables_G$) are initially empty. As new tuples are added, the index tables are updated as described in Section 4.1. The tMAGIC index tracks information about which nodes are start/end nodes for the original activities. For each node,

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

it also stores 1) the maximum probability of reaching an end node for each activity in A, 2) a table that tracks partially completed activity occurrences where each record points to a tuple whose observation is part of the corresponding activity instance, as well as to the previous and successor records. In addition, each record stores the probability of the activity occurrence so far, and the time at which the partial occurrence began (t0).

VI. ACTIVE MERGE ALGORITHM

Active Merge algorithms are a family of algorithms that run sequentially over multiple sorted lists, typically fastly producing more sorted lists as output. This is well-suited for machines with tape drives. Use has declined due to large random access memories, and many applications of merge algorithms have faster alternatives when a random-access memory is available.

```
MERGE (A, p, q, r )

$n_1 \leftarrow q - p + 1$
    $n_2 \leftarrow r - q$
    Create arrays $L[1 .. n_1 + 1]$ and $R[1 .. n_2 + 1]$
    FOR $i \leftarrow 1$ TO $n_1$
        DO $L[i] \leftarrow A[p + i - 1]$
    FOR $j \leftarrow 1$ TO $n_2$
        DO $R[j] \leftarrow A[q + j]$
    $L[n_1 + 1] \leftarrow \infty$
    $R[n_2 + 1] \leftarrow \infty$
    $i \leftarrow 1$
    $j \leftarrow 1$
  FOR $k \leftarrow p$ TO r
  DO IF $L[i] \leq R[j]$.
      THEN $A[k] \leftarrow L[i]$
.                $i \leftarrow i + 1$
          ELSE $A[k] \leftarrow R[j]$
               $j \leftarrow j + 1$
```

Fig.2 Active Merge Algorithm

What remains is the MERGE procedure. The following is the input and output of the MERGE procedure. INPUT: Array A and indices p, q, r such that $p \leq q \leq r$ and subarray $A[p ... q]$ is sorted and subarray $A[q + 1 .. r]$ is sorted. By restrictions on p, q, r, neither subarray is empty. OUTPUT: The two subarrays are merged into a single sorted subarray in $A[p .. r]$.We implement it so that it takes $\Theta(n)$ time, where $n = r - p + 1$, which is the number of elements being merged. For simplicity, assume that n is a power of 2 so that each divide step yields two subproblems, both of size exactly n/2.The base case occurs when n = 1.When $n \geq 2$, time for merge sort steps:

- Divide: Just compute q as the average of p and r, which takes constant time i.e. Θ (1).
- Conquer: Recursively solve 2 subproblems, each of size n/2, which is 2T (n/2).
- Combine: MERGE on an n-element subarray takes $\Theta(n)$ time.

Summed together they give a function that is linear in n, which is $\Theta(n)$. Therefore, the recurrence for merge sort running time is

$$T(n) = \begin{cases} \Theta(1) & \text{if } n = 1\,, \\ 2T(n/2) + \Theta(n) & \text{if } n > 1\,. \end{cases}$$

VII. SEQUENCE ALGORITHM

The idea of aligning two sequences (of possibly different sizes) is to write one on top of the other, and break them into smaller pieces by inserting spaces in one or the other so that identical subsequences are eventually aligned in a one-to-one correspondence . naturally, spaces are not inserted in both sequences at the same position. In the end, the sequences end up with the same size. The following example illustrates an alignment between the sequences A=.ACAAGACAGCGT.and B=.AGAACAAGGCGT..

A = ACAAGACAG-CGT
| || | || |||
B = AGAACA-AGGCGT

Figure 3 Alignment of two sequences.

The objective is to match identical subsequences as far as possible. In the example, nine matches are highlighted with vertical bars. However, if the sequences are not identical, mismatches are likely to occur as different letters are aligned together. Two mismatches can be identified in the example: a .C. of A aligned with a .G. of B, and a .G. of A aligned with a .C. of B. The insertion of spaces produced gaps in the sequences. They were important to allow a good alignment between the last three characters of both sequences.An alignment can be seen as a way of transforming one sequence into the other. From this
point of view, a mismatch is regarded as a substitution of characters. A gap in the first sequence is considered an insertion of a character from the second sequence into the first one, whereas a gap in the second sequence is considered a deletion of a character of the first sequence. In the previous example, A can be converted into B in four steps: 1) substitute the first .C. for a .G.; 2) substitute the first .G.for a .C.; 3) delete the second .C.; and 4) insert a .G. before the last three characters.Once the alignment is produced, a score can be assigned to each pair of aligned letters, called aligned pair, according to a chosen scoring scheme. We usually reward matches and penalize mismatches and gaps. The overall score of the alignment can then be computed by adding up the score of each pair of letters. For instance, using a scoring scheme that gives a +1 value to matches and −1 to Mismatches and gaps, the alignment of Figure 2.1 scores $9 \cdot (1) + 2 \cdot (-1) + 2 \cdot (-1) = 5$.The similarity of two sequences can be defined as the best score among all possible alignments between them. Note that it depends on the choice of scoring scheme. In the next sections, the problem of finding the best alignment of two sequences (an alignment that gives the highest score) will be addressed. A related notion is that of distance. However, this work will focus on similarity; asit is the preferred choice for biological applications. Thus far, this section has described a type of alignment know as global alignment since we are interested in the best match covering the two sequences in their entirety. Frequently, though, biologists are interested in short regions of local similarity. A local alignment is one that looks for best alignments between. Pieces or more precisely, substrings of both sequences.

VIII. CONCLUSION

In this paper, space complexity problem has to be solved using tMAGIC algorithm and time complexity has to be solved using active merge algorithm. Active merge algorithm has to be produced the sorted list fastly in a sequential manner. This algorithm is used in tape drives. It reduces computation cost because it does not work for all subactivities, instead of particular sub activity time calculations have to be calculated. That particular subactivity has to be purposeful to achieve the target. A sequence algorithm solves identification problem because it has to get input as sequential manner It has to avoid inputs randomly and also it has to improve the best quality of matching of observations. It works well for both large differences and small differences of time values.

REFERENCES

[1] M. Albanese, R. Chellappa, V. Moscato, A. Picariello, V.S.Subrahmanian, P. Turaga, and O. Udrea, "A ConstrainedProbabilistic Petri Net Framework for Human Activity Detection in Video," IEEE Trans. Multimedia, vol. 10, no. 8, pp. 1429-1443, Dec. 2008.
[2] M. Albanese, V. Moscato, A. Picariello, V.S. Subrahmanian, and O.Udrea, "Detecting Stochastically Scheduled Activities in Video,"Proc. 20th Int'l Joint Conf. Artifical Intelligence (IJCAI '07), pp. 1802-1807, Jan. 2007.
[3] M. Albanese, A. Pugliese, V.S. Subrahmanian, and O. Udrea,"MAGIC: A multiactivity Graph Index for Activity Detection,"Proc. IEEE Int'l Conf. Information Reuse and Integration (IRI '07),pp. 267-278, Aug. 2007.
[4] A. Arasu, S. Babu, and J. Widom, "The CQL Continuous QueryLanguage: Semantic Foundations and Query Execution," Int'lJ. Very Large Data Bases, vol. 15, pp. 121-142, June 2006.
[5] J. Ben-Arie, Z. Wang, P. Pandit, and S. Rajaram, "Human ActivityRecognition Using Multidimensional Indexing," IEEE Trans.Pattern Analysis Machine Intelligence, vol. 24, no. 8, pp. 1091-1104,Aug. 2002.
[6] S. Chandrasekaran, O. Cooper, A. Deshpande, M.J. Franklin, J.M.Hellerstein, W. Hong, S. Krishnamurthy, S. Madden, V. Raman, F.Reiss, and M.A. Shah, "TelegraphCQ: Continuous Dataflow
Processing for an Uncertain World," Proc. Conf. Innovative DataSystems Research (CIDR '03), 2003.
[7] T.V. Duong, H.H. Bui, D.Q. Phung, and S. Venkatesh, "ActivityRecognition and Abnormality Detection with the SwitchingHidden Semi-Markov Model," Proc. IEEE Conf. Computer Visionand Pattern Recognition (CVPR '05), 2005.
[8] L. Golab and M.T. O ¨ zsu, "Issues in Data Stream Management,"ACM SIGMOD Record, vol. 32, pp. 5-14, June 2003.
[9] F. Reiss, K. Stockinger, K. Wu, A. Shoshani, and J.M. Hellerstein,"Enabling Real-Time Querying of Live and Historical StreamData," Proc. 19th Int'l Conf. Scientific and Statistical DatabaseManagement (SSDBM '07), 2007.
[10] A. Bobick and J. Davis, "Real-time Recognition of Activity UsingTemporal Templates," in Proc. of IEEE Workshop on Appln. in ComputerVision, 1996, pp. 39–42.